Red Hat Enterprise Linux 6
DM Multipath

DM Multipath Configuration and Administration
Edition 1

Red Hat Enterprise Linux 6 DM Multipath

DM Multipath Configuration and Administration
Edition 1

Legal Notice

Copyright © 2014 Red Hat, Inc. and others.

This document is licensed by Red Hat under the Creative Commons Attribution-ShareAlike 3.0 Unported License. If you distribute this document, or a modified version of it, you must provide attribution to Red Hat, Inc. and provide a link to the original. If the document is modified, all Red Hat trademarks must be removed.

Red Hat, as the licensor of this document, waives the right to enforce, and agrees not to assert, Section 4d of CC-BY-SA to the fullest extent permitted by applicable law.

Red Hat, Red Hat Enterprise Linux, the Shadowman logo, JBoss, MetaMatrix, Fedora, the Infinity Logo, and RHCE are trademarks of Red Hat, Inc., registered in the United States and other countries.

Linux ® is the registered trademark of Linus Torvalds in the United States and other countries.

Java ® is a registered trademark of Oracle and/or its affiliates.

XFS ® is a trademark of Silicon Graphics International Corp. or its subsidiaries in the United States and/or other countries.

MySQL ® is a registered trademark of MySQL AB in the United States, the European Union and other countries.

Node.js ® is an official trademark of Joyent. Red Hat Software Collections is not formally related to or endorsed by the official Joyent Node.js open source or commercial project.

The OpenStack ® Word Mark and OpenStack Logo are either registered trademarks/service marks or trademarks/service marks of the OpenStack Foundation, in the United States and other countries and are used with the OpenStack Foundation's permission. We are not affiliated with, endorsed or sponsored by the OpenStack Foundation, or the OpenStack community.

All other trademarks are the property of their respective owners.

Abstract

This book provides information on using the Device-Mapper Multipath feature of Red Hat Enterprise Linux 6.

Table of Contents

Preface **3**
1. Audience 3
2. Related Documentation 3
3. We Need Feedback! 3
4. Document Conventions 4
4.1. Typographic Conventions 4
4.2. Pull-quote Conventions 5
4.3. Notes and Warnings 6

Chapter 1. Device Mapper Multipathing **7**
1.1. New and Changed Features 7
1.1.1. New and Changed Features for Red Hat Enterprise Linux 6.0 7
1.1.2. New and Changed Features for Red Hat Enterprise Linux 6.1 8
1.1.3. New and Changed Features for Red Hat Enterprise Linux 6.2 8
1.1.4. New and Changed Features for Red Hat Enterprise Linux 6.3 8
1.1.5. New and Changed Features for Red Hat Enterprise Linux 6.4 9
1.1.6. New and Changed Features for Red Hat Enterprise Linux 6.5 9
1.1.7. New and Changed Features for Red Hat Enterprise Linux 6.6 9
1.2. Overview of DM-Multipath 9
1.3. Storage Array Support 12
1.4. DM-Multipath Components 12
1.5. DM-Multipath Setup Overview 13

Chapter 2. Multipath Devices **14**
2.1. Multipath Device Identifiers 14
2.2. Consistent Multipath Device Names in a Cluster 14
2.3. Multipath Device Attributes 15
2.4. Multipath Devices in Logical Volumes 15

Chapter 3. Setting Up DM-Multipath **17**
3.1. Setting Up DM-Multipath 17
3.2. Ignoring Local Disks when Generating Multipath Devices 18
3.3. Configuring Storage Devices 20
3.4. Setting Up Multipathing in the initramfs File System 21

Chapter 4. The DM-Multipath Configuration File **22**
4.1. Configuration File Overview 22
4.2. Configuration File Blacklist 23
4.2.1. Blacklisting by WWID 24
4.2.2. Blacklisting By Device Name 24
4.2.3. Blacklisting By Device Type 25
4.2.4. Blacklist Exceptions 25
4.3. Configuration File Defaults 25
4.4. Multipaths Device Configuration Attributes 32
4.5. Configuration File Devices 35

Chapter 5. DM-Multipath Administration and Troubleshooting **40**
5.1. Resizing an Online Multipath Device 40
5.2. Moving root File Systems from a Single Path Device to a Multipath Device 40
5.3. Moving swap File Systems from a Single Path Device to a Multipath Device 43
5.4. The Multipath Daemon 44
5.5. Issues with Large Number of LUNs 44
5.6. Issues with queue_if_no_path feature 44

5.7. Multipath Command Output 45
5.8. Multipath Queries with multipath Command 45
5.9. Multipath Command Options 46
5.10. Determining Device Mapper Entries with the dmsetup Command 47
5.11. Troubleshooting with the multipathd Interactive Console 47

Revision History **49**

Index **51**

Preface

This book describes the Device Mapper Multipath (DM-Multipath) feature of Red Hat Enterprise Linux for the Red Hat Enterprise Linux 6 release.

1. Audience

This book is intended to be used by system administrators managing systems running the Linux operating system. It requires familiarity with Red Hat Enterprise Linux.

2. Related Documentation

For more information about using Red Hat Enterprise Linux, refer to the following resources:

- *Installation Guide* — Documents relevant information regarding the installation of Red Hat Enterprise Linux 6.
- *Deployment Guide* — Documents relevant information regarding the deployment, configuration and administration of Red Hat Enterprise Linux 6.
- *Storage Administration Guide* — Provides instructions on how to effectively manage storage devices and file systems on Red Hat Enterprise Linux 6.

For more information about Red Hat Cluster Suite for Red Hat Enterprise Linux 6, refer to the following resources:

- *High Availability Add-On Overview* — Provides a high-level overview of the Red Hat High Availability Add-On.
- *Cluster Administration* — Provides information about installing, configuring and managing the High Availability Add-On.
- *Logical Volume Manager Administration* — Provides a description of the Logical Volume Manager (LVM), including information on running LVM in a clustered environment.
- *Global File System 2: Configuration and Administration* — Provides information about installing, configuring, and maintaining Red Hat GFS2 (Red Hat Global File System 2).
- *Load Balancer Administration* — Provides information on configuring high-performance systems and services with the Load Balancer Add-On, a set of integrated software components that provide Linux Virtual Servers (LVS) for balancing IP load across a set of real servers.
- *Release Notes* — Provides information about the current release of Red Hat products.

Red Hat Cluster Suite documentation and other Red Hat documents are available in HTML, PDF, and RPM versions on the Red Hat Enterprise Linux Documentation CD and online at https://access.redhat.com/site/documentation/.

3. We Need Feedback!

If you find a typographical error in this manual, or if you have thought of a way to make this manual better, we would love to hear from you! Please submit a report in Bugzilla: http://bugzilla.redhat.com/ against the product **Red Hat Enterprise Linux 6** and the component **doc-DM_Multipath**. When submitting a bug report, be sure to mention the manual's identifier:

```
rh-DM_Multipath(EN)-6 (2014-10-8T15:15)
```

If you have a suggestion for improving the documentation, try to be as specific as possible when describing it. If you have found an error, please include the section number and some of the surrounding text so we can find it easily.

4. Document Conventions

This manual uses several conventions to highlight certain words and phrases and draw attention to specific pieces of information.

4.1. Typographic Conventions

Four typographic conventions are used to call attention to specific words and phrases. These conventions, and the circumstances they apply to, are as follows.

`Mono-spaced Bold`

Used to highlight system input, including shell commands, file names and paths. Also used to highlight keys and key combinations. For example:

> To see the contents of the file **`my_next_bestselling_novel`** in your current working directory, enter the **`cat my_next_bestselling_novel`** command at the shell prompt and press **`Enter`** to execute the command.

The above includes a file name, a shell command and a key, all presented in mono-spaced bold and all distinguishable thanks to context.

Key combinations can be distinguished from an individual key by the plus sign that connects each part of a key combination. For example:

> Press **`Enter`** to execute the command.
>
> Press **`Ctrl`**+**`Alt`**+**`F2`** to switch to a virtual terminal.

The first example highlights a particular key to press. The second example highlights a key combination: a set of three keys pressed simultaneously.

If source code is discussed, class names, methods, functions, variable names and returned values mentioned within a paragraph will be presented as above, in **`mono-spaced bold`**. For example:

> File-related classes include **`filesystem`** for file systems, **`file`** for files, and **`dir`** for directories. Each class has its own associated set of permissions.

Proportional Bold

This denotes words or phrases encountered on a system, including application names; dialog-box text; labeled buttons; check-box and radio-button labels; menu titles and submenu titles. For example:

> Choose **System** → **Preferences** → **Mouse** from the main menu bar to launch **Mouse Preferences**. In the **Buttons** tab, select the **`Left-handed mouse`** check box and click **`Close`** to switch the primary mouse button from the left to the right (making the mouse suitable for use in the left hand).
>
> To insert a special character into a **`gedit`** file, choose **Applications** → **Accessories** → **Character Map** from the main menu bar. Next, choose **Search** →

> **Find...** from the **Character Map** menu bar, type the name of the character in the **Search** field and click **Next**. The character you sought will be highlighted in the **Character Table**. Double-click this highlighted character to place it in the **Text to copy** field and then click the **Copy** button. Now switch back to your document and choose **Edit** → **Paste** from the **gedit** menu bar.

The above text includes application names; system-wide menu names and items; application-specific menu names; and buttons and text found within a GUI interface, all presented in proportional bold and all distinguishable by context.

Mono-spaced Bold Italic or ***Proportional Bold Italic***

Whether mono-spaced bold or proportional bold, the addition of italics indicates replaceable or variable text. Italics denotes text you do not input literally or displayed text that changes depending on circumstance. For example:

> To connect to a remote machine using ssh, type **ssh *username*@*domain.name*** at a shell prompt. If the remote machine is **example.com** and your username on that machine is john, type **ssh john@example.com**.
>
> The **mount -o remount *file-system*** command remounts the named file system. For example, to remount the **/home** file system, the command is **mount -o remount /home**.
>
> To see the version of a currently installed package, use the **rpm -q *package*** command. It will return a result as follows: ***package-version-release***.

Note the words in bold italics above: username, domain.name, file-system, package, version and release. Each word is a placeholder, either for text you enter when issuing a command or for text displayed by the system.

Aside from standard usage for presenting the title of a work, italics denotes the first use of a new and important term. For example:

> Publican is a *DocBook* publishing system.

4.2. Pull-quote Conventions

Terminal output and source code listings are set off visually from the surrounding text.

Output sent to a terminal is set in **mono-spaced roman** and presented thus:

```
books        Desktop   documentation  drafts  mss    photos   stuff  svn
books_tests  Desktop1  downloads      images  notes  scripts  svgs
```

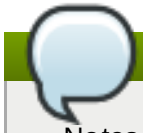
Source-code listings are also set in **mono-spaced roman** but add syntax highlighting as follows:

```
static int kvm_vm_ioctl_deassign_device(struct kvm *kvm,
                  struct kvm_assigned_pci_dev *assigned_dev)
{
         int r = 0;
         struct kvm_assigned_dev_kernel *match;

         mutex_lock(&kvm->lock);

         match = kvm_find_assigned_dev(&kvm->arch.assigned_dev_head,
                                       assigned_dev->assigned_dev_id);
```

```
        if (!match) {
                printk(KERN_INFO "%s: device hasn't been assigned
before, "
                  "so cannot be deassigned\n", __func__);
                r = -EINVAL;
                goto out;
        }

        kvm_deassign_device(kvm, match);

        kvm_free_assigned_device(kvm, match);

out:
        mutex_unlock(&kvm->lock);
        return r;
}
```

4.3. Notes and Warnings

Finally, we use three visual styles to draw attention to information that might otherwise be overlooked.



Note

Notes are tips, shortcuts or alternative approaches to the task at hand. Ignoring a note should have no negative consequences, but you might miss out on a trick that makes your life easier.

Important

Important boxes detail things that are easily missed: configuration changes that only apply to the current session, or services that need restarting before an update will apply. Ignoring a box labeled “Important” will not cause data loss but may cause irritation and frustration.

Warning

Warnings should not be ignored. Ignoring warnings will most likely cause data loss.

Chapter 1. Device Mapper Multipathing

Device mapper multipathing (DM-Multipath) allows you to configure multiple I/O paths between server nodes and storage arrays into a single device. These I/O paths are physical SAN connections that can include separate cables, switches, and controllers. Multipathing aggregates the I/O paths, creating a new device that consists of the aggregated paths.

This chapter provides a summary of the features of DM-Multipath that are new for the initial release of Red Hat Enterprise Linux 6. Following that, this chapter provides a high-level overview of DM Multipath and its components, as well as an overview of DM-Multipath setup.

1.1. New and Changed Features

This section lists new and changed features of DM-Multipath that are included with the initial and subsequent releases of Red Hat Enterprise Linux 6.

1.1.1. New and Changed Features for Red Hat Enterprise Linux 6.0

Red Hat Enterprise Linux 6.0 includes the following documentation and feature updates and changes.

- For the Red Hat Enterprise Linux 6 release, the initial DM-Multipath setup procedure for a basic failover configuration has changed. You can now create the DM-Multipath configuration file and enable DM-Multipath with the **mpathconf** configuration utility, which can also load the **device-mapper-multipath** module, start the **multipathd** daemon, and set **chkconfig** to start the daemon automatically on reboot.

  For information on the new setup procedure, see Section 3.1, “Setting Up DM-Multipath”. For more information on the **mpathconf** command, see the **mpathconf**(5) man page.

- The Red Hat Enterprise Linux 6 release provides a new mode for setting up multipath devices, which you set with the **find_multipaths** configuration file parameter. In previous releases of Red Hat Enterprise Linux, multipath always tried to create a multipath device for every path that was not explicitly blacklisted. In Red Hat Enterprise Linux 6, however, if the **find_multipaths** configuration parameter is set to **yes**, then multipath will create a device only if one of three conditions are met:

  - There are at least two non-blacklisted paths with the same WWID.
  - The user manually forces the device creation, by specifying a device with the **multipath** command.
  - A path has the same WWID as a multipath device that was previously created (even if that multipath device does not currently exist). For instructions on the procedure to follow if you have previously created multipath devices when the **find_multipaths** parameter was not set, see Section 4.2, “Configuration File Blacklist”.

  This feature should allow most users to have multipath automatically choose the correct paths to make into multipath devices, without having to edit the blacklist.

  For information on the **find_multipaths** configuration parameter, see Section 4.3, “Configuration File Defaults”.

- The Red Hat Enterprise Linux 6 release provides two new path selector algorithms which determine which path to use for the next I/O operation: **queue-length** and **service-time**. The **queue-length** algorithm looks at the amount of outstanding I/O to the paths to determine which

path to use next. The **service-time** algorithm looks at the amount of outstanding I/O and the relative throughput of the paths to determine which path to use next. For more information on the path selector parameters in the configuration file, see Chapter 4, *The DM-Multipath Configuration File*.

- In the Red Hat Enterprise Linux 6 release, priority functions are no longer callout programs. Instead they are dynamic shared objects like the path checker functions. The **prio_callout** parameter has been replaced by the **prio** parameter. For descriptions of the supported **prio** functions, see Chapter 4, *The DM-Multipath Configuration File*.
- In Red Hat Enterprise Linux 6, the **multipath** command output has changed format. For information on the **multipath** command output, see Section 5.7, "Multipath Command Output".
- In the Red Hat Enterprise Linux 6 release, the location of the multipath **bindings** file is **/etc/multipath/bindings**.
- The Red Hat Enterprise Linux 6 release provides three new **defaults** parameters in the **multipath.conf** file: **checker_timeout**, **fast_io_fail_tmo**, and **dev_loss_tmo**. For information on these parameters, see Chapter 4, *The DM-Multipath Configuration File*.
- When the **user_friendly_names** option in the multipath configuration file is set to **yes**, the name of a multipath device is of the form **mpath***n*. For the Red Hat Enterprise Linux 6 release, *n* is an alphabetic character, so that the name of a multipath device might be **mpatha** or **mpathb**. In previous releases, *n* was an integer.

1.1.2. New and Changed Features for Red Hat Enterprise Linux 6.1

Red Hat Enterprise Linux 6.1 includes the following documentation and feature updates and changes.

- This document now contains a new chapter, Section 5.2, "Moving root File Systems from a Single Path Device to a Multipath Device".
- This document now contains a new chapter, Section 5.3, "Moving swap File Systems from a Single Path Device to a Multipath Device".

1.1.3. New and Changed Features for Red Hat Enterprise Linux 6.2

Red Hat Enterprise Linux 6.2 includes the following documentation and feature updates and changes.

- The Red Hat Enterprise Linux 6.2 release provides a new **multipath.conf** parameter, **rr_min_io_rq**, in the **defaults**, **devices**, and **multipaths** sections of the **multipath.conf** file. The **rr_min_io** parameter no longer has an effect in Red Hat Enterprise Linux 6.2. For information on the **rr_min_io_rq** parameter, see Chapter 4, *The DM-Multipath Configuration File*.
- The **dev_loss_tmo** configuration file parameter can now be set to infinity, which sets the actual **sysfs** variable to 2147483647 seconds, or 68 years. For information on this parameter, see Chapter 4, *The DM-Multipath Configuration File*.
- The procedure described in Section 5.2, "Moving root File Systems from a Single Path Device to a Multipath Device" has been updated.

1.1.4. New and Changed Features for Red Hat Enterprise Linux 6.3

Red Hat Enterprise Linux 6.3 includes the following documentation and feature updates and changes.

- The default value of the **queue_without_daemon** configuration file parameter is now set to **no** by default.
- The default value of the **max_fds** configuration file parameter is now set to **max** by default.
- The **user_friendly_names** configuration file parameter is now configurable in the **defaults**, **multipaths**, and **devices** sections of the **multipath.conf** configuration file.
- The **defaults** section of the **multipath.conf** configuration file supports a new **hwtable_regex_match** parameter.

For information on the configuration file parameters, see Chapter 4, *The DM-Multipath Configuration File*.

1.1.5. New and Changed Features for Red Hat Enterprise Linux 6.4

Red Hat Enterprise Linux 6.4 includes the following documentation and feature updates and changes.

- The **defaults** section and the **devices** section of the **multipath.conf** configuration file support a new **retain_attached_hardware_handler** parameter and a new **detect_prio** parameter. For information on the configuration file parameters, see Chapter 4, *The DM-Multipath Configuration File*.
- This document contains a new section, Section 3.4, “Setting Up Multipathing in the initramfs File System”.

1.1.6. New and Changed Features for Red Hat Enterprise Linux 6.5

Red Hat Enterprise Linux 6.5 includes the following documentation and feature updates and changes.

- The **defaults** section of the **multipath.conf** configuration file supports a new **replace_wwide_whitespace** and a new **reload_rewrite** parameter. The **defaults** section of the **multipath.conf** file is documented in Table 4.1, “Multipath Configuration Defaults”.

1.1.7. New and Changed Features for Red Hat Enterprise Linux 6.6

Red Hat Enterprise Linux 6.6 includes the following documentation and feature updates and changes.

- The **defaults** section of the **multipath.conf** configuration file supports a new **force_sync** parameter. The **defaults** section of the **multipath.conf** file is documented in Table 4.1, “Multipath Configuration Defaults”.
- The **multipath** supports a **-w** and a **-W**, as described in Table 4.1, “Multipath Configuration Defaults”.

1.2. Overview of DM-Multipath

DM-Multipath can be used to provide:

- Redundancy

  DM-Multipath can provide failover in an active/passive configuration. In an active/passive configuration, only half the paths are used at any time for I/O. If any element of an I/O path (the cable, switch, or controller) fails, DM-Multipath switches to an alternate path.

- Improved Performance

  DM-Multipath can be configured in active/active mode, where I/O is spread over the paths in a round-robin fashion. In some configurations, DM-Multipath can detect loading on the I/O paths and dynamically re-balance the load.

Figure 1.1, “Active/Passive Multipath Configuration with One RAID Device” shows an active/passive configuration with two I/O paths from the server to a RAID device. There are 2 HBAs on the server, 2 SAN switches, and 2 RAID controllers.

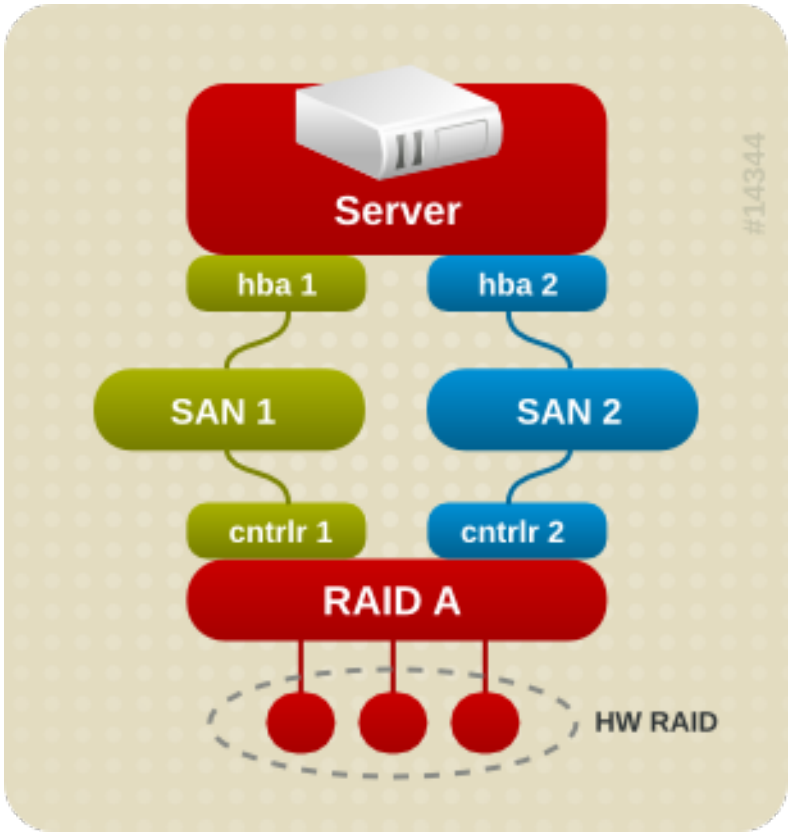


Figure 1.1. Active/Passive Multipath Configuration with One RAID Device

In this configuration, there is one I/O path that goes through hba1, SAN1, and controller 1 and a second I/O path that goes through hba2, SAN2, and controller2. There are many points of possible failure in this configuration:

- HBA failure
- FC cable failure
- SAN switch failure
- Array controller port failure

With DM-Multipath configured, a failure at any of these points will cause DM-Multipath to switch to the alternate I/O path.

Figure 1.2, “Active/Passive Multipath Configuration with Two RAID Devices” shows a more complex active/passive configuration with 2 HBAs on the server, 2 SAN switches, and 2 RAID devices with 2 RAID controllers each.

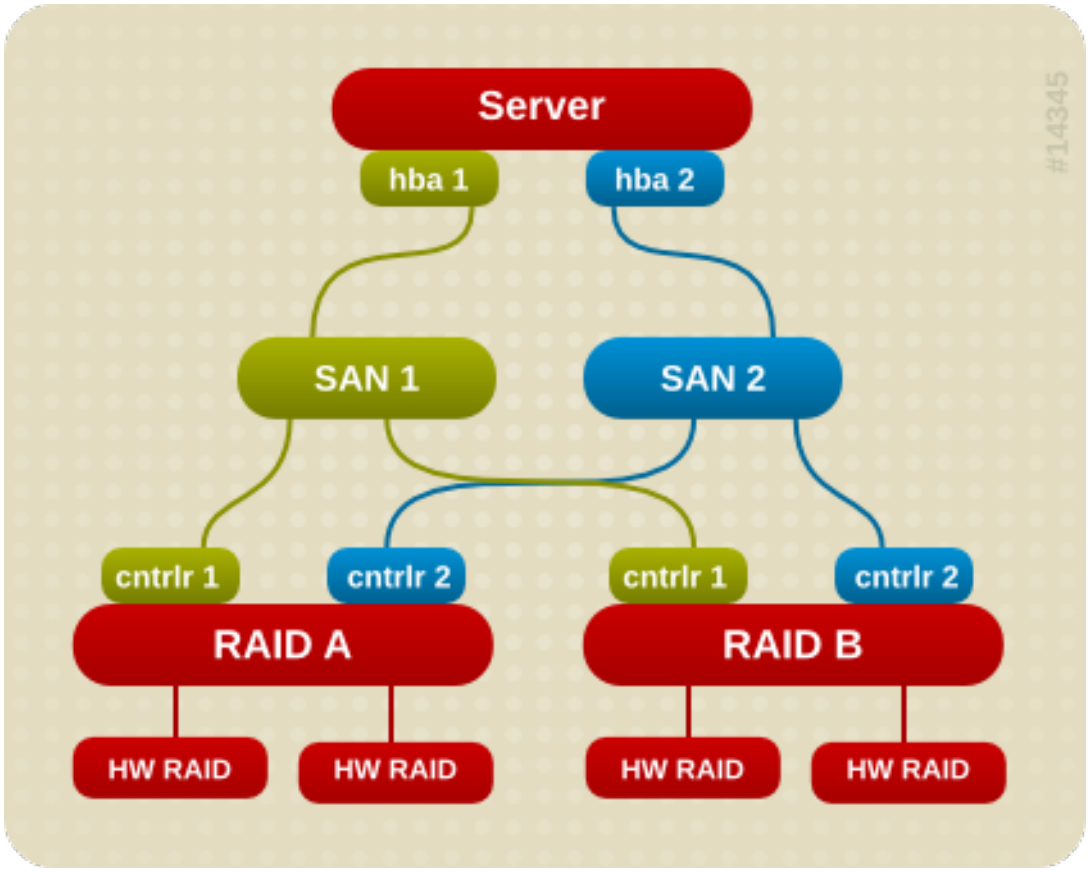


Figure 1.2. Active/Passive Multipath Configuration with Two RAID Devices

In the example shown in Figure 1.2, "Active/Passive Multipath Configuration with Two RAID Devices", there are two I/O paths to each RAID device (just as there are in the example shown in Figure 1.1, "Active/Passive Multipath Configuration with One RAID Device"). With DM-Multipath configured, a failure at any of the points of the I/O path to either of the RAID devices will cause DM-Multipath to switch to the alternate I/O path for that device.

Figure 1.3, "Active/Active Multipath Configuration with One RAID Device" shows an active/active configuration with 2 HBAs on the server, 1 SAN switch, and 2 RAID controllers. There are four I/O paths from the server to a storage device:

- hba1 to controller1
- hba1 to controller2
- hba2 to controller1
- hba2 to controller2

In this configuration, I/O can be spread among those four paths.

Figure 1.3. Active/Active Multipath Configuration with One RAID Device

1.3. Storage Array Support

By default, DM-Multipath includes support for the most common storage arrays that support DM-Multipath. The supported devices can be found in the **multipath.conf.defaults** file. If your storage array supports DM-Multipath and is not configured by default in this file, you may need to add them to the DM-Multipath configuration file, **multipath.conf**. For information on the DM-Multipath configuration file, see Chapter 4, *The DM-Multipath Configuration File*.

Some storage arrays require special handling of I/O errors and path switching. These require separate hardware handler kernel modules.

1.4. DM-Multipath Components

Table 1.1, "DM-Multipath Components". describes the components of DM-Multipath.

Table 1.1. DM-Multipath Components

Component	Description
dm_multipath kernel module	Reroutes I/O and supports failover for paths and path groups.
mpathconf utility	Configures and enables device mapper multipathing.
multipath command	Lists and configures multipath devices. Normally started up with **/etc/rc.sysinit**, it can also be started up by a **udev** program whenever a block device is added.
multipathd daemon	Monitors paths; as paths fail and come back, it may initiate path group switches. Provides for interactive changes to multipath devices. This must be restarted for any changes to the **/etc/multipath.conf** file.

Component	Description
kpartx command	Creates device mapper devices for the partitions on a device It is necessary to use this command for DOS-based partitions with DM-MP. The **kpartx** is provided in its own package, but the **device-mapper-multipath** package depends on it.

1.5. DM-Multipath Setup Overview

DM-Multipath includes compiled-in default settings that are suitable for common multipath configurations. Setting up DM-multipath is often a simple procedure.

The basic procedure for configuring your system with DM-Multipath is as follows:

1. Install the **device-mapper-multipath** rpm.
2. Create the configuration file and enable multipathing with the **mpathconf** command. You can also start the multipath daemon with this command if you do not need to edit the configuration file.
3. If necessary, edit the **multipath.conf** configuration file to modify default values and save the updated file.
4. Start the multipath daemon.

For detailed setup instructions for multipath configuration see Chapter 3, *Setting Up DM-Multipath*.

Chapter 2. Multipath Devices

Without DM-Multipath, each path from a server node to a storage controller is treated by the system as a separate device, even when the I/O path connects the same server node to the same storage controller. DM-Multipath provides a way of organizing the I/O paths logically, by creating a single multipath device on top of the underlying devices.

2.1. Multipath Device Identifiers

Each multipath device has a World Wide Identifier (WWID), which is guaranteed to be globally unique and unchanging. By default, the name of a multipath device is set to its WWID. Alternately, you can set the **user_friendly_names** option in the multipath configuration file, which sets the alias to a node-unique name of the form **mpath*n***.

For example, a node with two HBAs attached to a storage controller with two ports via a single unzoned FC switch sees four devices: **/dev/sda**, **/dev/sdb**, **dev/sdc**, and **/dev/sdd**. DM-Multipath creates a single device with a unique WWID that reroutes I/O to those four underlying devices according to the multipath configuration. When the **user_friendly_names** configuration option is set to **yes**, the name of the multipath device is set to **mpath*n***.

When new devices are brought under the control of DM-Multipath, the new devices may be seen in two different places under the **/dev** directory: **/dev/mapper/mpath*n*** and **/dev/dm-*n***.

- The devices in **/dev/mapper** are created early in the boot process. Use these devices to access the multipathed devices, for example when creating logical volumes.
- Any devices of the form **/dev/dm-*n*** are for internal use only and should never be used.

For information on the multipath configuration defaults, including the **user_friendly_names** configuration option, see Section 4.3, “Configuration File Defaults”.

You can also set the name of a multipath device to a name of your choosing by using the **alias** option in the **multipaths** section of the multipath configuration file. For information on the **multipaths** section of the multipath configuration file, see Section 4.4, “Multipaths Device Configuration Attributes”.

2.2. Consistent Multipath Device Names in a Cluster

When the **user_friendly_names** configuration option is set to **yes**, the name of the multipath device is unique to a node, but it is not guaranteed to be the same on all nodes using the multipath device. Similarly, if you set the **alias** option for a device in the **multipaths** section of the **multipath.conf** configuration file, the name is not automatically consistent across all nodes in the cluster. This should not cause any difficulties if you use LVM to create logical devices from the multipath device, but if you require that your multipath device names be consistent in every node it is recommended that you not set the **user_friendly_names** option to **yes** and that you not configure aliases for the devices. By default, if you do not set **user_friendly_names** to **yes** or configure an alias for a device, a device name will be the WWID for the device, which is always the same.

If you want the system-defined user-friendly names to be consistent across all nodes in the cluster, however, you can follow this procedure:

1. Set up all of the multipath devices on one machine.
2. Disable all of your multipath devices on your other machines by running the following commands:

```
# service multipathd stop
# multipath -F
```

3. Copy the **/etc/multipath/bindings** file from the first machine to all the other machines in the cluster.
4. Re-enable the **multipathd** daemon on all the other machines in the cluster by running the following command:

```
# service multipathd start
```

If you add a new device, you will need to repeat this process.

Similarly, if you configure an alias for a device that you would like to be consistent across the nodes in the cluster, you should ensure that the **/etc/multipath.conf** file is the same for each node in the cluster by following the same procedure:

1. Configure the aliases for the multipath devices in the in the **multipath.conf** file on one machine.
2. Disable all of your multipath devices on your other machines by running the following commands:

```
# service multipathd stop
# multipath -F
```

3. Copy the **/etc/multipath.conf** file from the first machine to all the other machines in the cluster.
4. Re-enable the **multipathd** daemon on all the other machines in the cluster by running the following command:

```
# service multipathd start
```

When you add a new device you will need to repeat this process.

2.3. Multipath Device Attributes

In addition to the **user_friendly_names** and **alias** options, a multipath device has numerous attributes. You can modify these attributes for a specific multipath device by creating an entry for that device in the **multipaths** section of the multipath configuration file. For information on the **multipaths** section of the multipath configuration file, see Section 4.4, “Multipaths Device Configuration Attributes”.

2.4. Multipath Devices in Logical Volumes

After creating multipath devices, you can use the multipath device names just as you would use a physical device name when creating an LVM physical volume. For example, if **/dev/mapper/mpatha** is the name of a multipath device, the following command will mark **/dev/mapper/mpatha** as a physical volume.

```
pvcreate /dev/mapper/mpatha
```

You can use the resulting LVM physical device when you create an LVM volume group just as you would use any other LVM physical device.

Note

If you attempt to create an LVM physical volume on a whole device on which you have configured partitions, the **pvcreate** command will fail. Note that the Anaconda and Kickstart installation programs create empty partition tables if you do not specify otherwise for every block device. If you wish to use the whole device rather than a partition, you must remove the existing partitions from the device. You can remove existing partitions with the **kpartx -d** and the **fdisk** commands. If your system has block devices that are greater that 2Tb, you can use the **parted** command to remove partitions.

When you create an LVM logical volume that uses active/passive multipath arrays as the underlying physical devices, you should include filters in the **lvm.conf** to exclude the disks that underlie the multipath devices. This is because if the array automatically changes the active path to the passive path when it receives I/O, multipath will failover and failback whenever LVM scans the passive path if these devices are not filtered. For active/passive arrays that require a command to make the passive path active, LVM prints a warning message when this occurs.

To filter all SCSI devices in the LVM configuration file (**lvm.conf**), include the following filter in the **devices** section of the file.

```
filter = [ "r/block/", "r/disk/", "r/sd.*/", "a/.*/" ]
```

Chapter 3. Setting Up DM-Multipath

This chapter provides step-by-step example procedures for configuring DM-Multipath. It includes the following procedures:

- Basic DM-Multipath setup
- Ignoring local disks
- Adding more devices to the configuration file
- Starting multipath in the **initramfs** file system

3.1. Setting Up DM-Multipath

Before setting up DM-Multipath on your system, ensure that your system has been updated and includes the **device-mapper-multipath** package.

You set up multipath with the **mpathconf** utility, which creates the multipath configuration file **/etc/multipath.conf**.

- If the **/etc/multipath.conf** file already exists, the **mpathconf** utility will edit it.
- If the **/etc/multipath.conf** file does not exist, the **mpathconf** utility will use the **/usr/share/doc/device-mapper-multipath-0.4.9/multipath.conf** file as the starting file.
- If the **/usr/share/doc/device-mapper-multipath-0.4.9/multipath.conf** file does not exist the **mpathconf** utility will create the **/etc/multipath.conf** file from scratch.

If you do not need to edit the **/etc/multipath.conf** file, you can set up DM-Multipath for a basic failover configuration by running the following command. This command enables the multipath configuration file and starts the **multipathd** daemon.

```
# mpathconf --enable --with_multipathd y
```

If you need to edit the **/etc/multipath.conf** file before starting the **multipathd** daemon. use the following procedure to set up DM-Multipath for a basic failover configuration.

1. Run the **mpathconf** command with the **--enable** option specified:

   ```
   # mpathconf --enable
   ```

   For information on additional options to the **mpathconf** command you may require, see the **mpathconf** man page or run the **mpathconf** command with the **--help** option specified.

   ```
   # mpathconf --help
   usage: /sbin/mpathconf <command>

   Commands:
   Enable: --enable
   Disable: --disable
   Set user_friendly_names (Default n): --user_friendly_names <y|n>
   Set find_multipaths (Default n): --find_multipaths <y|n>
   ```

```
Load the dm-multipath modules on enable (Default y): --with_module
<y|n>
start/stop/reload multipathd (Default n): --with_multipathd  <y|n>
chkconfig on/off multipathd (Default y): --with_chkconfig  <y|n>
```

2. Edit the **/etc/multipath.conf** file if necessary. The default settings for DM-Multipath are compiled in to the system and do not need to be explicitly set in the **/etc/multipath.conf** file.

   The default value of **path_grouping_policy** is set to **failover**, so in this example you do not need to edit the **/etc/multipath.conf** file. For information on changing the values in the configuration file to something other than the defaults, see Chapter 4, *The DM-Multipath Configuration File*.

   The initial defaults section of the configuration file configures your system so that the names of the multipath devices are of the form **mpath***n*; without this setting, the names of the multipath devices would be aliased to the WWID of the device.

3. Save the configuration file and exit the editor, if necessary.

4. Execute the following command:

```
# service multipathd start
```

Since the value of **user_friendly_name** is set to **yes** in the configuration file, the multipath devices will be created as **/dev/mapper/mpath***n*. For information on setting the name of the device to an alias of your choosing, see Chapter 4, *The DM-Multipath Configuration File*.

If you do not want to use user friendly names, you can run the following command:

```
# mpathconf --enable --user_friendly_names n
```

Note

If you find that you need to edit the multipath configuration file after you have started the multipath daemon, you must execute the **service multipathd reload** command for the changes to take effect.

3.2. Ignoring Local Disks when Generating Multipath Devices

Some machines have local SCSI cards for their internal disks. DM-Multipath is not recommended for these devices. If you set the **find_multipaths** configuration parameter to **yes**, you should not have to blacklist these devices. For information on the **find_multipaths** configuration parameter, see Section 4.3, “Configuration File Defaults”.

If you do not set the **find_multipaths** configuration parameter to **yes**, can use the following procedure to modify the multipath configuration file to ignore the local disks when configuring multipath.

1. Determine which disks are the internal disks and mark them as the ones to blacklist.

In this example, **/dev/sda** is the internal disk. Note that as originally configured in the default multipath configuration file, executing the **multipath -v2** shows the local disk, **/dev/sda**, in the multipath map.

For further information on the **multipath** command output, see Section 5.7, “Multipath Command Output”.

```
# multipath -v2
create: SIBM-ESXSST336732LC____F3ET0EP0Q000072428BX1 undef
WINSYS,SF2372
size=33 GB features="0" hwhandler="0" wp=undef
`-+- policy='round-robin 0' prio=1 status=undef
  |- 0:0:0:0 sda 8:0  [---------

device-mapper ioctl cmd 9 failed: Invalid argument
device-mapper ioctl cmd 14 failed: No such device or address
create: 3600a0b80001327d80000006d43621677 undef WINSYS,SF2372
size=12G features='0' hwhandler='0' wp=undef
`-+- policy='round-robin 0' prio=1 status=undef
  |- 2:0:0:0 sdb 8:16  undef ready  running
  `- 3:0:0:0 sdf 8:80 undef ready  running

create: 3600a0b800013275100000009a436215ec undef WINSYS,SF2372
size=12G features='0' hwhandler='0' wp=undef
`-+- policy='round-robin 0' prio=1 status=undef
  |- 2:0:0:1 sdc 8:32 undef ready  running
  `- 3:0:0:1 sdg 8:96 undef ready  running

create: 3600a0b80001327d800000070436216b3 undef WINSYS,SF2372
size=12G features='0' hwhandler='0' wp=undef
`-+- policy='round-robin 0' prio=1 status=undef
  |- 2:0:0:2 sdd 8:48 undef ready  running
  `- 3:0:0:2 sdg 8:112 undef ready  running

create: 3600a0b800013275100000009b4362163e undef WINSYS,SF2372
size=12G features='0' hwhandler='0' wp=undef
`-+- policy='round-robin 0' prio=1 status=undef
  |- 2:0:0:3 sdd 8:64 undef ready  running
  `- 3:0:0:3 sdg 8:128 undef ready  running
```

2. In order to prevent the device mapper from mapping **/dev/sda** in its multipath maps, edit the blacklist section of the **/etc/multipath.conf** file to include this device. Although you could blacklist the **sda** device using a **devnode** type, that would not be safe procedure since **/dev/sda** is not guaranteed to be the same on reboot. To blacklist individual devices, you can blacklist using the WWID of that device.

   Note that in the output to the **multipath -v2** command, the WWID of the **/dev/sda** device is SIBM-ESXSST336732LC____F3ET0EP0Q000072428BX1. To blacklist this device, include the following in the **/etc/multipath.conf** file.

```
blacklist {
      wwid SIBM-ESXSST336732LC____F3ET0EP0Q000072428BX1
}
```

3. After you have updated the **/etc/multipath.conf** file, you must manually tell the **multipathd** daemon to reload the file. The following command reloads the updated **/etc/multipath.conf** file.

```
# service multipathd reload
```

4. Run the following command to remove the multipath device:

```
# multipath -f SIBM-ESXSST336732LC____F3ET0EP0Q000072428BX1
```

5. To check whether the device removal worked, you can run the **multipath -ll** command to display the current multipath configuration. For information on the **multipath -ll** command, see Section 5.8, “Multipath Queries with multipath Command”.

   To check that the blacklisted device was not added back, you can run the **multipath** command, as in the following example. The **multipath** command defaults to a verbosity level of **v2** if you do not specify specify a **-v** option.

```
# multipath

create: 3600a0b80001327d80000006d43621677 undef WINSYS,SF2372
size=12G features='0' hwhandler='0' wp=undef
`-+- policy='round-robin 0' prio=1 status=undef
  |- 2:0:0:0 sdb 8:16  undef ready  running
  `- 3:0:0:0 sdf 8:80 undef ready  running

create: 3600a0b800013275100000009a436215ec undef WINSYS,SF2372
size=12G features='0' hwhandler='0' wp=undef
`-+- policy='round-robin 0' prio=1 status=undef
  |- 2:0:0:1 sdc 8:32 undef ready  running
  `- 3:0:0:1 sdg 8:96 undef ready  running

create: 3600a0b80001327d800000070436216b3 undef WINSYS,SF2372
size=12G features='0' hwhandler='0' wp=undef
`-+- policy='round-robin 0' prio=1 status=undef
  |- 2:0:0:2 sdd 8:48 undef ready  running
  `- 3:0:0:2 sdg 8:112 undef ready  running

create: 3600a0b800013275100000009b4362163e undef WINSYS,SF2372
size=12G features='0' hwhandler='0' wp=undef
`-+- policy='round-robin 0' prio=1 status=undef
  |- 2:0:0:3 sdd 8:64 undef ready  running
  `- 3:0:0:3 sdg 8:128 undef ready  running
```

3.3. Configuring Storage Devices

By default, DM-Multipath includes support for the most common storage arrays that support DM-Multipath. The default configuration values, including supported devices, can be found in the **multipath.conf.defaults** file.

If you need to add a storage device that is not supported by default as a known multipath device, edit the **/etc/multipath.conf** file and insert the appropriate device information.

For example, to add information about the HP Open-V series the entry looks like this, where %n is the device name:

```
devices {
     device {
            vendor "HP"
            product "OPEN-V."
            getuid_callout "/lib/udev/scsi_id --whitelisted --
device=/dev/%n"
     }
}
```

For more information on the **devices** section of the configuration file, see Section 4.5, “Configuration File Devices”.

3.4. Setting Up Multipathing in the initramfs File System

You can set up multipathing in the **initramfs** file system. After configuring multipath, you can rebuild the **initramfs** file system with the multipath configuration files by executing the **dracut** command with the following options:

```
# dracut --force --add multipath --include /etc/multipath /etc/multipath
```

If you run multipath from the **initramfs** file system and you make any changes to the multipath configuration files, you must rebuild the **initramfs** file system for the changes to take effect.

Chapter 4. The DM-Multipath Configuration File

By default, DM-Multipath provides configuration values for the most common uses of multipathing. In addition, DM-Multipath includes support for the most common storage arrays that support DM-Multipath. The default configuration values and the supported devices can be found in the **/usr/share/doc/device-mapper-multipath-0.4.9/multipath.conf.defaults** file.

You can override the default configuration values for DM-Multipath by editing the **/etc/multipath.conf** configuration file. If necessary, you can also add a storage array that is not supported by default to the configuration file.

Note

You can run set up multipathing in the **initramfs** file system. If you run multipath from the **initramfs** file system and you make any changes to the multipath configuration files, you must rebuild the **initramfs** file system for the changes to take effect. For information on rebuilding the **initramfs** file system with multipath, refer to Section 3.4, “Setting Up Multipathing in the initramfs File System”.

This chapter provides information on parsing and modifying the **multipath.conf** file. It contains sections on the following topics:

- Configuration file overview
- Configuration file blacklist
- Configuration file defaults
- Configuration file multipaths
- Configuration file devices

In the multipath configuration file, you need to specify only the sections that you need for your configuration, or that you wish to change from the default values specified in the **multipath.conf.defaults** file. If there are sections of the file that are not relevant to your environment or for which you do not need to override the default values, you can leave them commented out, as they are in the initial file.

The configuration file allows regular expression description syntax.

An annotated version of the configuration file can be found in **/usr/share/doc/device-mapper-multipath-0.4.9/multipath.conf.annotated**.

4.1. Configuration File Overview

The multipath configuration file is divided into the following sections:

blacklist

Listing of specific devices that will not be considered for multipath.

blacklist_exceptions

Listing of multipath candidates that would otherwise be blacklisted according to the parameters of the blacklist section.

defaults

General default settings for DM-Multipath.

multipaths

Settings for the characteristics of individual multipath devices. These values overwrite what is specified in the **defaults** and **devices** sections of the configuration file.

devices

Settings for the individual storage controllers. These values overwrite what is specified in the **defaults** section of the configuration file. If you are using a storage array that is not supported by default, you may need to create a **devices** subsection for your array.

When the system determines the attributes of a multipath device, first it checks the multipath settings, then the per devices settings, then the multipath system defaults.

4.2. Configuration File Blacklist

The **blacklist** section of the multipath configuration file specifies the devices that will not be used when the system configures multipath devices. Devices that are blacklisted will not be grouped into a multipath device.

In older releases of Red Hat Enterprise Linux, multipath always tried to create a multipath device for every path that was not explicitly blacklisted. In Red Hat Enterprise Linux 6, however, if the **find_multipaths** configuration parameter is set to **yes**, then multipath will create a device only if one of three conditions are met:

- There are at least two non-blacklisted paths with the same WWID.
- The user manually forces the creation of the device by specifying a device with the **multipath** command.
- A path has the same WWID as a multipath device that was previously created (even if that multipath device does not currently exist). Whenever a multipath device is created, multipath remembers the WWID of the device so that it will automatically create the device again as soon as it sees a path with that WWID. This allows you to have multipath automatically choose the correct paths to make into multipath devices, without have to edit the multipath blacklist.

  If you have previously created a multipath device without using the **find_multipaths** parameter and then you later set the parameter to **yes**, you may need to remove the WWIDs of any device you do not want created as a multipath device from the **/etc/multipath/wwids** file. The following shows a sample **/etc/multipath/wwids** file. The WWIDs are enclosed by slashes (/):

```
# Multipath wwids, Version : 1.0
# NOTE: This file is automatically maintained by multipath and
multipathd.
# You should not need to edit this file in normal circumstances.
#
# Valid WWIDs:
/3600d0230000000000e13955cc3757802/
/3600d0230000000000e13955cc3757801/
/3600d0230000000000e13955cc3757800/
/3600d02300069c9ce09d41c31f29d4c00/
/SWINSYS  SF2372         0E13955CC3757802/
/3600d0230000000000e13955cc3757803/
```

With the **find_multipaths** parameter set to **yes**, you need to blacklist only the devices with multiple paths that you do not want to be multipathed. Because of this, it will generally not be necessary to blacklist devices.

If you do need to blacklist devices, you can do so according to the following criteria:

- By WWID, as described in Section 4.2.1, “Blacklisting by WWID”
- By device name, as described in Section 4.2.2, “Blacklisting By Device Name”
- By device type, as described in Section 4.2.3, “Blacklisting By Device Type”

By default, a variety of device types are blacklisted, even after you comment out the initial blacklist section of the configuration file. For information, see Section 4.2.2, “Blacklisting By Device Name”.

4.2.1. Blacklisting by WWID

You can specify individual devices to blacklist by their World-Wide IDentification with a **wwid** entry in the **blacklist** section of the configuration file.

The following example shows the lines in the configuration file that would blacklist a device with a WWID of 26353900f02796769.

```
blacklist {
       wwid 26353900f02796769
}
```

4.2.2. Blacklisting By Device Name

You can blacklist device types by device name so that they will not be grouped into a multipath device by specifying a **devnode** entry in the **blacklist** section of the configuration file.

The following example shows the lines in the configuration file that would blacklist all SCSI devices, since it blacklists all sd* devices.

```
blacklist {
       devnode "^sd[a-z]"
}
```

You can use a **devnode** entry in the **blacklist** section of the configuration file to specify individual devices to blacklist rather than all devices of a specific type. This is not recommended, however, since unless it is statically mapped by **udev** rules, there is no guarantee that a specific device will have the same name on reboot. For example, a device name could change from **/dev/sda** to **/dev/sdb** on reboot.

By default, the following **devnode** entries are compiled in the default blacklist; the devices that these entries blacklist do not generally support DM-Multipath. To enable multipathing on any of these devices, you would need to specify them in the **blacklist_exceptions** section of the configuration file, as described in Section 4.2.4, “Blacklist Exceptions”.

```
blacklist {
       devnode "^(ram|raw|loop|fd|md|dm-|sr|scd|st)[0-9]*"
       devnode "^(td|hd)[a-z]"
}
```

4.2.3. Blacklisting By Device Type

4.2.3. Blacklisting By Device Type

You can specify specific device types in the **blacklist** section of the configuration file with a **device** section. The following example blacklists all IBM DS4200 and HP devices.

```
blacklist {
       device {
               vendor  "IBM"
               product "3S42"       #DS4200 Product 10
       }
       device {
               vendor  "HP"
               product "*"
       }
}
```

4.2.4. Blacklist Exceptions

You can use the **blacklist_exceptions** section of the configuration file to enable multipathing on devices that have been blacklisted by default.

For example, if you have a large number of devices and want to multipath only one of them (with the WWID of 3600d0230000000000e13955cc3757803), instead of individually blacklisting each of the devices except the one you want, you could instead blacklist all of them, and then allow only the one you want by adding the following lines to the **/etc/multipath.conf** file.

```
blacklist {
        wwid "*"
}

blacklist_exceptions {
        wwid "3600d0230000000000e13955cc3757803"
}
```

When specifying devices in the **blacklist_exceptions** section of the configuration file, you must specify the exceptions in the same way they were specified in the blacklist. For example, a WWID exception will not apply to devices specified by a **devnode** blacklist entry, even if the blacklisted device is associated with that WWID. Similarly, **devnode** exceptions apply only to **devnode** entries, and **device** exceptions apply only to device entries.

4.3. Configuration File Defaults

The **/etc/multipath.conf** configuration file includes a **defaults** section that sets the **user_friendly_names** parameter to **yes**, as follows.

```
defaults {
        user_friendly_names yes
}
```

This overwrites the default value of the **user_friendly_names** parameter.

The configuration file includes a template of configuration defaults. This section is commented out, as follows.

```
#defaults {
#       udev_dir                /dev
#       polling_interval        5
#       path_selector           "round-robin 0"
#       path_grouping_policy    failover
#       getuid_callout          "/lib/udev/scsi_id --whitelisted --
device=/dev/%n"
# prio   const
# path_checker  directio
# rr_min_io  1000
# rr_weight  uniform
# failback  manual
# no_path_retry  fail
# user_friendly_names no
#}
```

To overwrite the default value for any of the configuration parameters, you can copy the relevant line from this template into the **defaults** section and uncomment it. For example, to overwrite the **path_grouping_policy** parameter so that it is **multibus** rather than the default value of **failover**, copy the appropriate line from the template to the initial **defaults** section of the configuration file, and uncomment it, as follows.

```
defaults {
        user_friendly_names     yes
        path_grouping_policy    multibus
}
```

Table 4.1, “Multipath Configuration Defaults” describes the attributes that are set in the **defaults** section of the **multipath.conf** configuration file. These values are used by DM-Multipath unless they are overwritten by the attributes specified in the **devices** and **multipaths** sections of the **multipath.conf** file.

Note

As of the Red Hat Enterprise Linux 6.0 release, the **mode**, **uid**, and **gid** parameters have been deprecated. Permissions for device-mapper devices (including mulitpath mappings) are set by means of **udev** rules. There is a template file in **/usr/share/doc/device-mapper-*version*** called **12-dm-permissions.rules** which you can use and place in the **/etc/udev/rules.d** directory for it to take effect.

Table 4.1. Multipath Configuration Defaults

Attribute	Description
polling_interval	Specifies the interval between two path checks in seconds. For properly functioning paths, the interval between checks will gradually increase to (4 * **polling_interval**). The default value is 5.
udev_dir	The directory where **udev** device nodes are created. The default value is **/dev**.
multipath_dir	The directory where the dynamic shared objects are stored. The default value is system dependent, commonly **/lib/multipath**.

Attribute	Description
find_multipaths	Defines the mode for setting up multipath devices. If this parameter is set to **yes**, then multipath will not try to create a device for every non-blacklisted path. Instead multipath will create a device only if one of three conditions are met: - There are at least two non-blacklisted paths with the same WWID. - The user manually forces the creation of the device by specifying a device with the **multipath** command. - A path has the same WWID as a multipath device that was previously created. Whenever a multipath device is created with **find_multipaths** set, multipath remembers the WWID of the device so that it will automatically create the device again as soon as it sees a path with that WWID. This allows you to have multipath automatically choose the correct paths to make into multipath devices, without having to edit the multipath blacklist. For instructions on the procedure to follow if you have previously created multipath devices when the **find_multipaths** parameter was not set, see Section 4.2, "Configuration File Blacklist". The default value is **no**.
verbosity	The default verbosity. Higher values increase the verbosity level. Valid levels are between 0 and 6. The default value is **2**.
path_selector	Specifies the default algorithm to use in determining what path to use for the next I/O operation. Possible values include: **round-robin 0**: Loop through every path in the path group, sending the same amount of I/O to each. **queue-length 0**: Send the next bunch of I/O down the path with the least number of outstanding I/O requests. **service-time 0**: Send the next bunch of I/O down the path with the shortest estimated service time, which is determined by dividing the total size of the outstanding I/O to each path by its relative throughput. The default value is **round-robin 0**.

Attribute	Description
path_grouping_policy	Specifies the default path grouping policy to apply to unspecified multipaths. Possible values include: **failover**: 1 path per priority group. **multibus**: all valid paths in 1 priority group. **group_by_serial**: 1 priority group per detected serial number. **group_by_prio**: 1 priority group per path priority value. Priorities are determined by callout programs specified as global, per-controller, or per-multipath options. **group_by_node_name**: 1 priority group per target node name. Target node names are fetched in **/sys/class/fc_transport/target*/node_name**. The default value is **failover**.
getuid_callout	Specifies the default program and arguments to call out to obtain a unique path identifier. An absolute path is required. The default value is **/lib/udev/scsi_id --whitelisted --device=/dev/%n**.
prio	Specifies the default function to call to obtain a path priority value. For example, the ALUA bits in SPC-3 provide an exploitable **prio** value. Possible values include: **const**: Set a priority of 1 to all paths. **emc**: Generate the path priority for EMC arrays. **alua**: Generate the path priority based on the SCSI-3 ALUA settings. **tpg_pref**: Generate the path priority based on the SCSI-3 ALUA settings, using the preferred port bit. **ontap**: Generate the path priority for NetApp arrays. **rdac**: Generate the path priority for LSI/Engenio RDAC controller. **hp_sw**: Generate the path priority for Compaq/HP controller in active/standby mode. **hds**: Generate the path priority for Hitachi HDS Modular storage arrays. The default value is **const**.

Attribute	Description
features	The default extra features of multipath devices, using the format: "*number_of_features_plus_arguments feature1* ...". Possible values for **features** include: **queue_if_no_path**, which is the same as setting **no_path_retry** to **queue**. For information on issues that may arise when using this feature, see Section 5.6, "Issues with queue_if_no_path feature". **retain_attached_hw_handler**: (Red Hat Enterprise Linux Release 6.4 and later) If this parameter is set to **yes** and the scsi layer has already attached a hardware handler to the path device, multipath will not force the device to use the **hardware_handler** specified by the **multipath.conf** file. If the scsi layer has not attached a hardware handler, multipath will continue to use its configured hardware handler as usual. The default value is **no**. **pg_init_retries *n***: Retry path group initialization up to *n* times before failing where 1 <= *n* <= 50. **pg_init_delay_msecs *n***: Wait *n* milliseconds between path group initialization retries where 0 <= *n* <= 60000.
path_checker	Specifies the default method used to determine the state of the paths. Possible values include: **readsector0**: Read the first sector of the device. **tur**: Issue a **TEST UNIT READY** to the device. **emc_clariion**: Query the EMC Clariion specific EVPD page 0xC0 to determine the path. **hp_sw**: Check the path state for HP storage arrays with Active/Standby firmware. **rdac**: Check the path stat for LSI/Engenio RDAC storage controller. **directio**: Read the first sector with direct I/O. The default value is **directio**.

Attribute	Description
failback	Manages path group failback. A value of **immediate** specifies immediate failback to the highest priority path group that contains active paths. A value of **manual** specifies that there should not be immediate failback but that failback can happen only with operator intervention. A value of **followover** specifies that automatic failback should be performed when the first path of a path group becomes active. This keeps a node from automatically failing back when another node requested the failover. A numeric value greater than zero specifies deferred failback, expressed in seconds. The default value is **manual**.
rr_min_io	Specifies the number of I/O requests to route to a path before switching to the next path in the current path group. This setting is only for systems running kernels older than 2.6.31. Newer systems should use **rr_min_io_rq**. The default value is 1000.
rr_min_io_rq	Specifies the number of I/O requests to route to a path before switching to the next path in the current path group, using request-based device-mapper-multipath. This setting should be used on systems running current kernels. On systems running kernels older than 2.6.31, use **rr_min_io**. The default value is 1.
rr_weight	If set to **priorities**, then instead of sending **rr_min_io** requests to a path before calling **path_selector** to choose the next path, the number of requests to send is determined by **rr_min_io** times the path's priority, as determined by the **prio** function. If set to **uniform**, all path weights are equal. The default value is **uniform**.
no_path_retry	A numeric value for this attribute specifies the number of times the system should attempt to use a failed path before disabling queueing. A value of **fail** indicates immediate failure, without queueing. A value of **queue** indicates that queueing should not stop until the path is fixed. The default value is 0.
user_friendly_names	If set to **yes**, specifies that the system should use the **/etc/multipath/bindings** file to assign a persistent and unique alias to the multipath, in the form of **mpath*n***. If set to **no**, specifies that the system should use the WWID as the alias for the multipath. In either case, what is specified here will be overridden by any device-specific aliases you specify in the **multipaths** section of the configuration file. The default value is **no**.
queue_without_daemon	If set to **no**, the **multipathd** daemon will disable queueing for all devices when it is shut down. The default value is **no**.

Attribute	Description
flush_on_last_del	If set to **yes**, the **multipathd** daemon will disable queueing when the last path to a device has been deleted. The default value is **no**.
max_fds	Sets the maximum number of open file descriptors that can be opened by multipath and the **multipathd** daemon. This is equivalent to the **ulimit -n** command. As of the Red Hat Enterprise Linux 6.3 release, the default value is **max**, which sets this to the system limit from **/proc/sys/fs/nr_open**. For earlier releases, if this is not set the maximum number of open file descriptors is taken from the calling process; it is usually 1024. To be safe, this should be set to the maximum number of paths plus 32, if that number is greater than 1024.
checker_timeout	The timeout to use for prioritizers and path checkers that issue SCSI commands with an explicit timeout, in seconds. The default value is taken from **sys/block/sd*x*/device/timeout**.
fast_io_fail_tmo	The number of seconds the SCSI layer will wait after a problem has been detected on an FC remote port before failing I/O to devices on that remote port. This value should be smaller than the value of **dev_loss_tmo**. Setting this to **off** will disable the timeout. The default value is determined by the OS.
dev_loss_tmo	The number of seconds the SCSI layer will wait after a problem has been detected on an FC remote port before removing it from the system. Setting this to infinity will set this to 2147483647 seconds, or 68 years. The default value is determined by the OS.
hwtable_regex_match	(Red Hat Enterprise Linux Release 6.3 and later) Controls how multipath integrates the device configurations from the **devices** section of the configuration file with the built-in device configurations. Each device configuration in the **devices** section of the **multipath.conf** file will either create its own device configuration or it will modify one of the built-in device configurations. Prior to Red Hat Enterprise Linux 6.3, if the vendor, product, and revision strings in a user's device configuration exactly matched those strings in a built-in device configuration, the built-in configuration was modified by the options in the user's configuration. Otherwise. the user's device configuration was treated as a new configuration. If **hwtable_regex_match** is set to **yes**, a regular expression match is used instead. The vendor, product, and revision strings are all regular expressions. The user device configuration values for these options are matched against the built-in device configuration values. This match works the same way that an actual device's vendor product and revision strings are matched against a device configuration's strings to see which configuration should be used for the device. If the user's device configuration matches, then the built-in configuration is modified by the options in the user's configuration. Otherwise the user's device configuration is treated as a new configuration.

Attribute	Description
retain_attached_hw_handler	(Red Hat Enterprise Linux Release 6.4 and later) If this parameter is set to **yes** and the scsi layer has already attached a hardware handler to the path device, multipath will not force the device to use the **hardware_handler** specified by the **multipath.conf** file. If the scsi layer has not attached a hardware handler, multipath will continue to use its configured hardware handler as usual. The default value is **no**.
detect_prio	(Red Hat Enterprise Linux Release 6.4 and later) If this is set to **yes**, multipath will first check if the device supports ALUA, and if so it will automatically assign the device the **alua** prioritizer. If the device doesn't support ALUA, it will determine the prioritizer as it always does. The default value is **no**.
replace_wwid_whitespace	(Red Hat Enterprise Linux Release 6.5 and later) If this is set to **yes**, the default **getuid_callout** for all devices that use this parameter will be changed to include **--replace-whitespace**.
reload_readwrite	(Red Hat Enterprise Linux Release 6.5 and later) If this is set to **yes**, the **multipathd** daemon will listen to path device change events and if a device has become read/write it will reload the multipath device.
force_sync	(Red Hat Enterprise Linux Release 6.6 and later) If this is set to "yes", it prevents path checkers from running in async mode.

4.4. Multipaths Device Configuration Attributes

Table 4.2, “Multipath Attributes” shows the attributes that you can set in the **multipaths** section of the **multipath.conf** configuration file for each specific multipath device. These attributes apply only to the one specified multipath. These defaults are used by DM-Multipath and override attributes set in the **defaults** and **devices** sections of the **multipath.conf** file.

Table 4.2. Multipath Attributes

Attribute	Description
wwid	Specifies the WWID of the multipath device to which the **multipath** attributes apply. This parameter is mandatory for this section of the **multipath.conf** file.
alias	Specifies the symbolic name for the multipath device to which the **multipath** attributes apply. If you are using **user_friendly_names**, do not set this value to **mpath*n***; this may conflict with an automatically assigned user friendly name and give you incorrect device node names.
path_grouping_policy	Specifies the default path grouping policy to apply to unspecified multipaths. Possible values include: **failover** = 1 path per priority group **multibus** = all valid paths in 1 priority group **group_by_serial** = 1 priority group per detected serial number **group_by_prio** = 1 priority group per path priority value **group_by_node_name** = 1 priority group per target node name

Attribute	Description
path_selector	Specifies the default algorithm to use in determining what path to use for the next I/O operation. Possible values include: **round-robin 0**: Loop through every path in the path group, sending the same amount of I/O to each. **queue-length 0**: Send the next bunch of I/O down the path with the least number of outstanding I/O requests. **service-time 0**: Send the next bunch of I/O down the path with the shortest estimated service time, which is determined by dividing the total size of the outstanding I/O to each path by its relative throughput.
failback	Manages path group failback. A value of **immediate** specifies immediate failback to the highest priority path group that contains active paths. A value of **manual** specifies that there should not be immediate failback but that failback can happen only with operator intervention. A value of **followover** specifies that automatic failback should be performed when the first path of a path group becomes active. This keeps a node from automatically failing back when another node requested the failover. A numeric value greater than zero specifies deferred failback, expressed in seconds.
prio	Specifies the default function to call to obtain a path priority value. For example, the ALUA bits in SPC-3 provide an exploitable **prio** value. Possible values include: **const**: Set a priority of 1 to all paths. **emc**: Generate the path priority for EMC arrays. **alua**: Generate the path priority based on the SCSI-3 ALUA settings. **tpg_pref**: Generate the path priority based on the SCSI-3 ALUA settings, using the preferred port bit. **ontap**: Generate the path priority for NetApp arrays. **rdac**: Generate the path priority for LSI/Engenio RDAC controller. **hp_sw**: Generate the path priority for Compaq/HP controller in active/standby mode. **hds**: Generate the path priority for Hitachi HDS Modular storage arrays.

Attribute	Description
no_path_retry	A numeric value for this attribute specifies the number of times the system should attempt to use a failed path before disabling queueing. A value of **fail** indicates immediate failure, without queueing. A value of **queue** indicates that queueing should not stop until the path is fixed.
rr_min_io	Specifies the number of I/O requests to route to a path before switching to the next path in the current path group. This setting is only for systems running kernels older that 2.6.31. Newer systems should use **rr_min_io_rq**. The default value is 1000.
rr_min_io_rq	Specifies the number of I/O requests to route to a path before switching to the next path in the current path group, using request-based device-mapper-multipath. This setting should be used on systems running current kernels. On systems running kernels older than 2.6.31, use **rr_min_io**. The default value is 1.
rr_weight	If set to **priorities**, then instead of sending **rr_min_io** requests to a path before calling **path_selector** to choose the next path, the number of requests to send is determined by **rr_min_io** times the path's priority, as determined by the **prio** function. If set to **uniform**, all path weights are equal.
flush_on_last_del	If set to **yes**, then multipath will disable queueing when the last path to a device has been deleted.
user_friendly_names	If set to **yes**, specifies that the system should use the **/etc/multipath/bindings** file to assign a persistent and unique alias to the multipath, in the form of **mpath*n***. If set to **no**, specifies that the system should use use the WWID as the alias for the multipath. In either case, what is specified here will be overridden by any device-specific aliases you specify in the **multipaths** section of the configuration file.

The following example shows multipath attributes specified in the configuration file for two specific multipath devices. The first device has a WWID of **3600508b4000156d70001200000b0000** and a symbolic name of **yellow**.

The second multipath device in the example has a WWID of **1DEC_____321816758474** and a symbolic name of **red**. In this example, the **rr_weight** attributes is set to **priorities**.

```
multipaths {
       multipath {
              wwid                    3600508b4000156d70001200000b0000
              alias                   yellow
              path_grouping_policy    multibus
              path_selector           "round-robin 0"
              failback                manual
              rr_weight               priorities
              no_path_retry           5
       }
       multipath {
              wwid                    1DEC_____321816758474
```

```
                alias                   red
                rr_weight               priorities
        }
}
```

4.5. Configuration File Devices

Table 4.3, “Device Attributes” shows the attributes that you can set for each individual storage device in the **devices** section of the **multipath.conf** configuration file. These attributes are used by DM-Multipath unless they are overwritten by the attributes specified in the **multipaths** section of the **multipath.conf** file for paths that contain the device. These attributes override the attributes set in the **defaults** section of the **multipath.conf** file.

Many devices that support multipathing are included by default in a multipath configuration. The values for the devices that are supported by default are listed in the **multipath.conf.defaults** file. You probably will not need to modify the values for these devices, but if you do you can overwrite the default values by including an entry in the configuration file for the device that overwrites those values. You can copy the device configuration defaults from the **multipath.conf.defaults** file for the device and override the values that you want to change.

To add a device that is not configured automatically by default to this section of the configuration file, you need to set the **vendor** and **product** parameters. You can find these values by looking at **/sys/block/*device_name*/device/vendor** and **/sys/block/*device_name*/device/model** where *device_name* is the device to be multipathed, as in the following example:

```
# cat /sys/block/sda/device/vendor
WINSYS
# cat /sys/block/sda/device/model
SF2372
```

The additional parameters to specify depend on your specific device. If the device is active/active, you will usually not need to set additional parameters. You may want to set **path_grouping_policy** to **multibus**. Other parameters you may need to set are **no_path_retry** and **rr_min_io**, as described in Table 4.3, “Device Attributes”.

If the device is active/passive, but it automatically switches paths with I/O to the passive path, you need to change the checker function to one that does not send I/O to the path to test if it is working (otherwise, your device will keep failing over). This almost always means that you set the **path_checker** to **tur**; this works for all SCSI devices that support the Test Unit Ready command, which most do.

Table 4.3. Device Attributes

Attribute	Description
vendor	Specifies the vendor name of the storage device to which the **device** attributes apply, for example **COMPAQ**.
product	Specifies the product name of the storage device to which the **device** attributes apply, for example **HSV110 (C)COMPAQ**.
revision	Specifies the product revision identifier of the storage device.
product_blacklist	Specifies a regular expression used to blacklist devices by product.

Attribute	Description
hardware_handler	Specifies a module that will be used to perform hardware specific actions when switching path groups or handling I/O errors. Possible values include: **1 emc**: hardware handler for EMC storage arrays. **1 alua**: hardware handler for SCSI-3 ALUA arrays. **1 hp_sw**: hardware handler for Compaq/HP controllers. **1 rdac**: hardware handler for the LSI/Engenio RDAC controllers.
path_grouping_policy	Specifies the default path grouping policy to apply to unspecified multipaths. Possible values include: **failover** = 1 path per priority group **multibus** = all valid paths in 1 priority group **group_by_serial** = 1 priority group per detected serial number **group_by_prio** = 1 priority group per path priority value **group_by_node_name** = 1 priority group per target node name
getuid_callout	Specifies the default program and arguments to call out to obtain a unique path identifier. An absolute path is required.
path_selector	Specifies the default algorithm to use in determining what path to use for the next I/O operation. Possible values include: **round-robin 0**: Loop through every path in the path group, sending the same amount of I/O to each. **queue-length 0**: Send the next bunch of I/O down the path with the least number of outstanding I/O requests. **service-time 0**: Send the next bunch of I/O down the path with the shortest estimated service time, which is determined by dividing the total size of the outstanding I/O to each path by its relative throughput.

Attribute	Description
path_checker	Specifies the default method used to determine the state of the paths. Possible values include: **readsector0**: Read the first sector of the device. **tur**: Issue a **TEST UNIT READY** to the device. **emc_clariion**: Query the EMC Clariion specific EVPD page 0xC0 to determine the path. **hp_sw**: Check the path state for HP storage arrays with Active/Standby firmware. **rdac**: Check the path stat for LSI/Engenio RDAC storage controller. **directio**: Read the first sector with direct I/O.
features	The default extra features of multipath devices, using the format: "*number_of_features_plus_arguments feature1* ...". Possible values for **features** include: **queue_if_no_path**, which is the same as setting **no_path_retry** to **queue**. For information on issues that may arise when using this feature, see Section 5.6, “Issues with queue_if_no_path feature”. **retain_attached_hw_handler**: (Red Hat Enterprise Linux Release 6.4 and later) If this parameter is set to **yes** and the scsi layer has already attached a hardware handler to the path device, multipath will not force the device to use the **hardware_handler** specified by the **multipath.conf** file. If the scsi layer has not attached a hardware handler, multipath will continue to use its configured hardware handler as usual. **pg_init_retries *n***: Retry path group initialization up to *n* times before failing where 1 <= *n* <= 50. **pg_init_delay_msecs *n***: Wait *n* milliseconds between path group initialization retries where 0 <= *n* <= 60000.

Attribute	Description
prio	Specifies the default function to call to obtain a path priority value. For example, the ALUA bits in SPC-3 provide an exploitable **prio** value. Possible values include: **const**: Set a priority of 1 to all paths. **emc**: Generate the path priority for EMC arrays. **alua**: Generate the path priority based on the SCSI-3 ALUA settings. **tpg_pref**: Generate the path priority based on the SCSI-3 ALUA settings, using the preferred port bit. **ontap**: Generate the path priority for NetApp arrays. **rdac**: Generate the path priority for LSI/Engenio RDAC controller. **hp_sw**: Generate the path priority for Compaq/HP controller in active/standby mode. **hds**: Generate the path priority for Hitachi HDS Modular storage arrays.
failback	Manages path group failback. A value of **immediate** specifies immediate failback to the highest priority path group that contains active paths. A value of **manual** specifies that there should not be immediate failback but that failback can happen only with operator intervention. A value of **followover** specifies that automatic failback should be performed when the first path of a path group becomes active. This keeps a node from automatically failing back when another node requested the failover. A numeric value greater than zero specifies deferred failback, expressed in seconds.
rr_weight	If set to **priorities**, then instead of sending **rr_min_io** requests to a path before calling **path_selector** to choose the next path, the number of requests to send is determined by **rr_min_io** times the path's priority, as determined by the **prio** function. If set to **uniform**, all path weights are equal.
no_path_retry	A numeric value for this attribute specifies the number of times the system should attempt to use a failed path before disabling queueing. A value of **fail** indicates immediate failure, without queueing. A value of **queue** indicates that queueing should not stop until the path is fixed.

Attribute	Description
rr_min_io	Specifies the number of I/O requests to route to a path before switching to the next path in the current path group. This setting is only for systems running kernels older that 2.6.31. Newer systems should use **rr_min_io_rq**. The default value is 1000.
rr_min_io_rq	Specifies the number of I/O requests to route to a path before switching to the next path in the current path group, using request-based device-mapper-multipath. This setting should be used on systems running current kernels. On systems running kernels older than 2.6.31, use **rr_min_io**. The default value is 1.
fast_io_fail_tmo	The number of seconds the SCSI layer will wait after a problem has been detected on an FC remote port before failing I/O to devices on that remote port. This value should be smaller than the value of **dev_loss_tmo**. Setting this to **off** will disable the timeout.
dev_loss_tmo	The number of seconds the SCSI layer will wait after a problem has been detected on an FC remote port before removing it from the system. Setting this to infinity will set this to 2147483647 seconds,or 68 years.
flush_on_last_del	If set to **yes**, the **multipathd** daemon will disable queueing when the last path to a device has been deleted.
user_friendly_names	If set to **yes**, specifies that the system should use the **/etc/multipath/bindings** file to assign a persistent and unique alias to the multipath, in the form of **mpath*n***. If set to **no**, specifies that the system should use use the WWID as the alias for the multipath. In either case, what is specified here will be overridden by any device-specific aliases you specify in the **multipaths** section of the configuration file. The default value is **no**.
retain_attached_hw_handler	(Red Hat Enterprise Linux Release 6.4 and later) If this parameter is set to **yes** and the scsi layer has already attached a hardware handler to the path device, multipath will not force the device to use the **hardware_handler** specified by the **multipath.conf** file. If the scsi layer has not attached a hardware handler, multipath will continue to use its configured hardware handler as usual.
detect_prio	(Red Hat Enterprise Linux Release 6.4 and later) If this is set to **yes**, multipath will first check if the device supports ALUA, and if so it will automatically assign the device the **alua** prioritizer. If the device doesn't support ALUA, it will determine the prioritizer as it always does.

The following example shows a **device** entry in the multipath configuration file.

```
# }
# device {
#  vendor   "COMPAQ  "
#  product   "MSA1000         "
#  path_grouping_policy multibus
#  path_checker  tur
#  rr_weight  priorities
# }
#}
```

Chapter 5. DM-Multipath Administration and Troubleshooting

This chapter will provide information on administering DM-Multipath on a running system. It includes sections on the following topics:

- Resizing an online multipath device
- Moving the root device from a single-path device to a multipath device
- Moving the swap device from a single path device to a multipath device
- The multipath daemon
- Issues with large number of LUNs
- Issues with queue_if_no_path feature
- **multipath** command output
- Multipath queries with the **multipath** command
- **multipath** command options
- Multipath queries with the **dmsetup** command
- Troubleshooting with the **multipathd** interactive console

5.1. Resizing an Online Multipath Device

If you need to resize an online multipath device, use the following procedure.

1. Resize your physical device.
2. Execute the following command to find the paths to the LUN:

```
# multipath -l
```

3. Resize your paths. For SCSI devices, writing a 1 to the **rescan** file for the device causes the SCSI driver to rescan, as in the following command:

```
# echo 1 > /sys/block/device_name/device/rescan
```

4. Resize your multipath device by executing the **multipathd resize** command:

```
# multipathd -k'resize map mpatha'
```

5. Resize the file system (assuming no LVM or DOS partitions are used):

```
# resize2fs /dev/mapper/mpatha
```

5.2. Moving root File Systems from a Single Path Device to a Multipath Device

If you have installed your system on a single-path device and later add another path to the root file system, you will need to move your root file system to a multipathed device. This section documents the procedure for moving from a single-path to a multipathed device.

After ensuring that you have installed the **device-mapper-multipath** package, perform the following procedure:

1. Execute the following command to create the **/etc/multipath.conf** configuration file, load the multipath module, and set **chkconfig** for the **multipathd** to **on**:

   ```
   # mpathconf --enable
   ```

   For further information on using the **mpathconf** command to set up multipathing, see Section 3.1, “Setting Up DM-Multipath”.

2. Edit the **blacklist** and **blacklist_exceptions** sections of the **/etc/multipath.conf** file, as described in Section 4.2, “Configuration File Blacklist”.

3. To confirm that your configuration file is set up correctly, you can run the **/sbin/multipath** command with the **-v3** option to check whether the multipath daemon tried to create a multipath device over your root device. The command will fail since the root the device is in use, but the output from the command should show the root device in the paths list.

   You should look in the command output for a line of the following format:

   ```
   WWID  H:B:T:L devname MAJOR:MINOR
   ```

   For example, if your root file system is set up on **sda** or one of its partitions, you would see a line in the output such as the following:

   ```
   ===== paths list =====
   ...
   1ATA     WDC WD800JD-75MSA3                              WD-WMAM9F
   1:0:0:0 sda 8:0
   ...
   ```

   Later in the output, you should see the root device assigned to a multipath device:

   ```
   time | devname: ownership set to mpathdev
   ```

   For example, the output may appear as follows:

   ```
   Jun 14 06:48:21 | sda: ownership set to mpatha
   ```

   You will also see an indication that the command failed to create the multipath device with a line of the following format:

   ```
   time | mpathdev: domap (0) failure for create/reload map
   ```

   In the example noted above, you would see the following line in the command output:

   ```
   Jun 14 06:48:21 | mpatha: domap (0) failure for create/reload map
   ```

4. To rebuild the **initramfs** file system with **multipath**, execute the **dracut** command with the following options:

```
# dracut --force --add multipath --include /etc/multipath
/etc/multipath
```

5. If your root device is not an LVM volume and it is mounted by device name, you may need to edit the **fstab** file to switch to the appropriate multipath device name. If your root device is an LVM device or is mounted by UUID or something else, this step is not necessary.

    a. Use the procedure described in Step 3 of running the **/sbin/multipath** command with the **-v3** to determine the WWID of the root device.

    b. Set up an alias for the root device in the **/etc/multipath.conf** file:

```
multipaths {
    multipath {
        wwid WWID_of_root_device
        alias rootdev
    }
}
```

    c. Edit the **/etc/fstab** and replace the old device path to the root device with the multipath device.

    For example, if you had the following entry in the **/etc/fstab** file:

```
/dev/sda1 /                       ext4    defaults        1 1
```

    You would change the entry to the following:

```
/dev/mapper/rootdev /             ext4    defaults        1 1
```

If you need to edit the **/etc/fstab** file, you will also need to edit the **/etc/grub/grub.conf** file and change the root parameter from **root=/dev/sda1** to **root=/dev/mapper/rootdev**.

The following example shows what this **grub.conf** file entry would look like before you edit it.

```
title Red Hat Enterprise Linux FoundationServer (2.6.32-
71.24.1.el6.x86_64)
        root (hd0,0)
        kernel /vmlinuz-2.6.32-71.24.1.el6.x86_64 ro root=/dev/sda1
rd_NO_LUKS
rd_NO_MD rd_NO_DM LANG=en_US.UTF-8 SYSFONT=latarcyrheb-sun16
KEYTABLE=us
console=ttyS0,115200n8 crashkernel=auto
        initrd /initramfs-2.6.32-71.24.1.el6.x86_64.img
```

The following example shows what the **grub.conf** file entry would look like after you edit it.

```
title Red Hat Enterprise Linux FoundationServer (2.6.32-
71.24.1.el6.x86_64)
        root (hd0,0)
```

```
        kernel /vmlinuz-2.6.32-71.24.1.el6.x86_64 ro
root=/dev/mapper/rootdev
rd_NO_LUKS rd_NO_MD rd_NO_DM LANG=en_US.UTF-8 SYSFONT=latarcyrheb-
sun16
KEYTABLE=us console=ttyS0,115200n8 crashkernel=auto
        initrd /initramfs-2.6.32-71.24.1.el6.x86_64.img
```

6. Shut the machine down.
7. Configure the FC switch so that other paths are visible to the machine.
8. Boot the machine.
9. Check whether the root file system ('/') is on the multipathed device.

5.3. Moving swap File Systems from a Single Path Device to a Multipath Device

By default, swap devices are set up as logical volumes. This does not require any special procedure for configuring them as multipath devices as long as you set up multipathing on the physical volumes that constitute the logical volume group. If your swap device is not an LVM volume, however, and it is mounted by device name, you may need to edit the **fstab** file to switch to the appropriate multipath device name.

1. Determine the WWID number of the swap device by running the **/sbin/multipath** command with the **-v3** option. The output from the command should show the swap device in the paths list.

   You should look in the command output for a line of the following format, showing the swap device:

   ```
   WWID  H:B:T:L devname MAJOR:MINOR
   ```

   For example, if your swap file system is set up on **sda** or one of its partitions, you would see a line in the output such as the following:

   ```
   ===== paths list =====
   ...
   1ATA     WDC WD800JD-75MSA3                                WD-WMAM9F
   1:0:0:0 sda 8:0
   ...
   ```

2. Set up an alias for the swap device in the **/etc/multipath.conf** file:

   ```
   multipaths {
       multipath {
           wwid WWID_of_swap_device
           alias swapdev
       }
   }
   ```

3. Edit the **/etc/fstab** and replace the old device path to the root device with the multipath device.

   For example, if you had the following entry in the **/etc/fstab** file:

```
/dev/sda2 swap                    ext4    defaults        0 0
```

You would change the entry to the following:

```
/dev/mapper/swapdev swap          ext4    defaults        0 0
```

5.4. The Multipath Daemon

If you find you have trouble implementing a multipath configuration, you should ensure that the multipath daemon is running, as described in Chapter 3, *Setting Up DM-Multipath*. The **multipathd** daemon must be running in order to use multipathed devices.

5.5. Issues with Large Number of LUNs

When a large number of LUNs are added to a node, using multipathed devices can significantly increase the time it takes for the **udev** device manager to create device nodes for them. If you experience this problem, you can correct it by deleting the following line in **/etc/udev/rules.d/40-multipath.rules**:

```
KERNEL!="dm-[0-9]*", ACTION=="add", PROGRAM=="/bin/bash -c '/sbin/lsmod |
/bin/grep ^dm_multipath'", RUN+="/sbin/multipath -v0 %M:%m"
```

This line causes the **udev** device manager to run **multipath** every time a block device is added to the node. Even with this line removed, the **multipathd** daemon will still automatically create multipathed devices, and **multipath** will still be called during the boot process for nodes with multipathed root file systems. The only change is that multipathed devices will not be automatically created when the **multipathd** daemon is not running, which should not be a problem for the vast majority of multipath users.

5.6. Issues with queue_if_no_path feature

If **features "1 queue_if_no_path"** is specified in the **/etc/multipath.conf** file, then any process that issues I/O will hang until one or more paths are restored. To avoid this, set the **no_path_retry *N*** parameter in the **/etc/multipath.conf** file (where ***N*** is the number of times the system should retry a path).

When you set the **no_path_retry** parameter, remove the **features "1 queue_if_no_path"** option from the **/etc/multipath.conf** file as well. If, however, you are using a multipathed device for which the **features "1 queue_if_no_path"** option is set as a compiled-in default, as it is for many SAN devices, you must explicitly add **features "0"** to override this default. You can do this by copying the existing devices section for your device from **/usr/share/doc/device-mapper-multipath-0.4.9/multipath.conf.defaults** into **/etc/multipath.conf** and editing it to suit your needs.

If you need to use the **features "1 queue_if_no_path"** option and you experience the issue noted here, use the **dmsetup** command to edit the policy at runtime for a particular LUN (that is, for which all the paths are unavailable). For example, if you want to change the policy on the multipath device **mpathc** from **"queue_if_no_path"** to **"fail_if_no_path"**, execute the following command.

```
dmsetup message mpathc 0 "fail_if_no_path"
```

Note that you must specify the **mpath*n*** alias rather than the path.

5.7. Multipath Command Output

When you create, modify, or list a multipath device, you get a display of the current device setup. The format is as follows.

For each multipath device:

```
 action_if_any: alias (wwid_if_different_from_alias)
dm_device_name_if_known vendor,product size=size features='features'
hwhandler='hardware_handler' wp=write_permission_if_known
```

For each path group:

```
-+- policy='scheduling_policy' prio=prio_if_known
status=path_group_status_if_known
```

For each path:

```
 `- host:channel:id:lun devnode major:minor dm_status_if_known
path_status online_status
```

For example, the output of a multipath command might appear as follows:

```
3600d0230000000000e13955cc3757800 dm-1 WINSYS,SF2372
size=269G features='0' hwhandler='0' wp=rw
|-+- policy='round-robin 0' prio=1 status=active
| `- 6:0:0:0 sdb 8:16  active ready  running
`-+- policy='round-robin 0' prio=1 status=enabled
  `- 7:0:0:0 sdf 8:80  active ready  running
```

If the path is up and ready for I/O, the status of the path is **ready** or **ghost**. If the path is down, the status is **faulty** or **shaky**. The path status is updated periodically by the **multipathd** daemon based on the polling interval defined in the **/etc/multipath.conf** file.

The dm status is similar to the path status, but from the kernel's point of view. The dm status has two states: **failed**, which is analogous to **faulty**, and **active** which covers all other path states. Occasionally, the path state and the dm state of a device will temporarily not agree.

The possible values for ***online_status*** are **running** and **offline**. A status of **offline** means that this SCSI device has been disabled.

Note

When a multipath device is being created or modified, the path group status, the dm device name, the write permissions, and the dm status are not known. Also, the features are not always correct.

5.8. Multipath Queries with multipath Command

You can use the **-l** and **-ll** options of the **multipath** command to display the current multipath configuration. The **-l** option displays multipath topology gathered from information in **sysfs** and the device mapper. The **-ll** option displays the information the **-l** displays in addition to all other available components of the system.

When displaying the multipath configuration, there are three verbosity levels you can specify with the **-v** option of the **multipath** command. Specifying **-v0** yields no output. Specifying **-v1** outputs the created or updated multipath names only, which you can then feed to other tools such as **kpartx**. Specifying **-v2** prints all detected paths, multipaths, and device maps.

The following example shows the output of a **multipath -l** command.

```
# multipath -l
3600d0230000000000e13955cc3757800 dm-1 WINSYS,SF2372
size=269G features='0' hwhandler='0' wp=rw
|-+- policy='round-robin 0' prio=1 status=active
| `- 6:0:0:0 sdb 8:16  active ready  running
`-+- policy='round-robin 0' prio=1 status=enabled
  `- 7:0:0:0 sdf 8:80  active ready  running
```

The following example shows the output of a **multipath -ll** command.

```
# multipath -ll
3600d0230000000000e13955cc3757801 dm-10 WINSYS,SF2372
size=269G features='0' hwhandler='0' wp=rw
|-+- policy='round-robin 0' prio=1 status=enabled
| `- 19:0:0:1 sdc 8:32  active ready  running
`-+- policy='round-robin 0' prio=1 status=enabled
  `- 18:0:0:1 sdh 8:112 active ready  running
3600d0230000000000e13955cc3757803 dm-2 WINSYS,SF2372
size=125G features='0' hwhandler='0' wp=rw
`-+- policy='round-robin 0' prio=1 status=active
  |- 19:0:0:3 sde 8:64  active ready  running
  `- 18:0:0:3 sdj 8:144 active ready  running
```

5.9. Multipath Command Options

Table 5.1, “Useful **multipath** Command Options” describes some options of the **multipath** command that you may find useful.

Table 5.1. Useful multipath Command Options

Option	Description
-l	Display the current multipath configuration gathered from **sysfs** and the device mapper.
-ll	Display the current multipath configuration gathered from **sysfs**, the device mapper, and all other available components on the system.
-f *device*	Remove the named multipath device.
-F	Remove all unused multipath devices.
-w *device*	(Red Hat Enterprise Linux 6.6 and later) Remove the **wwid** of the specified device from the **wwids** file.
-W	(Red Hat Enterprise Linux 6.6 and later) Reset the **wwids** file to include only the current multipath devices.

5.10. Determining Device Mapper Entries with the dmsetup Command

You can use the **dmsetup** command to find out which device mapper entries match the multipathed devices.

The following command displays all the device mapper devices and their major and minor numbers. The minor numbers determine the name of the dm device. For example, a minor number of 3 corresponds to the multipathed device **/dev/dm-3**.

```
# dmsetup ls
mpathd  (253:4)
mpathep1        (253:12)
mpathfp1        (253:11)
mpathb  (253:3)
mpathgp1        (253:14)
mpathhp1        (253:13)
mpatha  (253:2)
mpathh  (253:9)
mpathg  (253:8)
VolGroup00-LogVol01     (253:1)
mpathf  (253:7)
VolGroup00-LogVol00     (253:0)
mpathe  (253:6)
mpathbp1        (253:10)
mpathd  (253:5)
```

5.11. Troubleshooting with the multipathd Interactive Console

The **multipathd -k** command is an interactive interface to the **multipathd** daemon. Executing this command brings up an interactive multipath console. After executing this command, you can enter **help** to get a list of available commands, you can enter an interactive command, or you can enter **CTRL-D** to quit.

The **multipathd** interactive console can be used to troubleshoot problems you may be having with your system. For example, the following command sequence displays the multipath configuration, including the defaults, before exiting the console.

```
# multipathd -k
> > show config
> > CTRL-D
```

The following command sequence ensures that multipath has picked up any changes to the **multipath.conf**,

```
# multipathd -k
> > reconfigure
> > CTRL-D
```

Use the following command sequence to ensure that the path checker is working properly.

```
# multipathd -k
> > show paths
> > CTRL-D
```

Revision History

Revision 7.0-4 **Thu Aug 7 2014** **Steven Levine**
Version for 6.6 GA release

Revision 7.0-3 **Thu Aug 7 2014** **Steven Levine**
Version for 6.6 Beta release

Revision 7.0-1 **Mon Jul 7 2014** **Steven Levine**
Resolves: #1117038
Documents new **-w** and **-W** options of the **multipath** command.
Resolves: #1116946
Documents all possible values of **features** entry in **multipath.conf** file.
Resolves: #969080
Small editorial clarifications.
Resolves: #1081264
Documents **force_sync** parameter.

Revision 6.0-7 **Wed Apr 10 2014** **Steven Levine**
Administrative Update to fix Review Server Beta build

Revision 6.0-6 **Wed Nov 13 2013** **Steven Levine**
Version for 6.5 GA release

Revision 6.0-5 **Fri Sep 27 2013** **Steven Levine**
Version for 6.5 Beta release

Revision 6.0-3 **Fri Sep 27 2013** **Steven Levine**
Resolves: #987144, #987671
Documents new **reload_rewrite** and **replace_wwid_whitespace** parameters in the **multipath.conf** file.

Resolves: #911630, #969022
Corrects small typographical errors.

Revision 6.0-1 **Tue Sep 10 2013** **Steven Levine**
Adding new default parameters for RHEL 6.5.

Revision 5.0-9 **Mon Feb 18 2013** **Steven Levine**
Version for 6.4 GA release

Revision 5.0-7 **Mon Nov 26 2012** **Steven Levine**
Version for 6.4 Beta release

Revision 5.0-6 **Tue Nov 13 2012** **Steven Levine**

Resolves: #856829
Documents new **retain_attached_hardware_handler** and **detect_prio** configuration parameters.

Resolves: #815925
Documents the procedure for setting up multipathing in the **initramfs** file system.

Resolves: #836333
Corrects small typographical error.

Resolves: #856826
Updates sample **multipath.conf** excerpt to reflect updated **path_selector** parameter.

Revision 4.0-3 Fri Jun 15 2012 Steven Levine
Version for 6.3 GA release

Revision 4.0-2 Wed Apr 25 2012 Steven Levine
Resolves: #815925
Updates output format for **dmsetup ls** example.

Revision 4.0-1 Wed Mar 28 2012 Steven Levine
Resolves: #786399
Fixes small typographic error.

Resolves: #771420
Documents new DM-Multipath features for Red Hat Enterprise Linux 6.3.

Revision 3.0-3 Thu Dec 1 2011 Steven Levine
Release for GA of Red Hat Enterprise Linux 6.2

Resolves: #753899
Corrects description of **multipath -F** option.

Revision 3.0-2 Fri Oct 7 2011 Steven Levine
Resolves: #743767
Fixes typos and clarifies small issues.

Revision 3.0-1 Mon Sep 19 2011 Steven Levine
Initial revision for Red Hat Enterprise Linux 6.2 Beta release

Resolves: #707638
Documents new DM-Multipath features for Red Hat Enterprise Linux 6.2.

Resolves: #715457
Corrects filter example for SCSI devices.

Resolves: #623450
Updates procedures for moving root device to a multipathed volume.

Resolves: #725374, #738051
Corrects minor typographical errors.

Revision 2.0-1 **Thu May 19 2011** **Steven Levine**

Initial revision for Red Hat Enterprise Linux 6.1

Resolves: #623450
Adds new procedures for moving root and swap devices from single path to multipathed devices.

Resolves: #693948
Corrects small errors in tables of device attributes.

Resolves: #694683
Corrects small typographic errors.

Resolves: #702721
Removes outdated reference to /dev/mpath.

Revision 1.0-1 **Wed Nov 10 2010** **Steven Levine**

First version for the Red Hat Enterprise Linux 6 release

Index

Symbols

/etc/multipath.conf package, Setting Up DM-Multipath

A

active/active configuration
- definition, Overview of DM-Multipath
- illustration, Overview of DM-Multipath

active/passive configuration
- definition, Overview of DM-Multipath
- illustration, Overview of DM-Multipath

alias parameter , Multipaths Device Configuration Attributes
- configuration file, Multipath Device Identifiers

B

blacklist
- configuration file, Configuration File Blacklist
- default devices, Blacklisting By Device Name
- device name, Blacklisting By Device Name
- device type, Blacklisting By Device Type
- WWID, Blacklisting by WWID

blacklist_exceptions section
- multipath.conf file, Blacklist Exceptions

C

checker_timeout parameter, Configuration File Defaults

configuration file
- alias parameter, Multipaths Device Configuration Attributes
- blacklist, Configuration File Blacklist

- checker_timeout parameter, Configuration File Defaults
- detect_prio parameter, Configuration File Defaults, Multipaths Device Configuration Attributes
- dev_loss_tmo parameter, Configuration File Defaults, Configuration File Devices
- failback parameter, Configuration File Defaults, Multipaths Device Configuration Attributes, Configuration File Devices
- fast_io_fail_tmo parameter, Configuration File Defaults, Configuration File Devices
- features parameter, Configuration File Defaults, Configuration File Devices
- flush_on_last_del parameter, Configuration File Defaults, Multipaths Device Configuration Attributes, Configuration File Devices
- force_sync parameter, Configuration File Defaults
- getuid_callout parameter, Configuration File Defaults, Configuration File Devices
- hardware_handler parameter, Configuration File Devices
- hwtable_regex_match parameter, Configuration File Defaults
- max_fds parameter, Configuration File Defaults
- no_path_retry parameter, Configuration File Defaults, Multipaths Device Configuration Attributes, Configuration File Devices
- overview, Configuration File Overview
- path_checker parameter, Configuration File Defaults, Configuration File Devices
- path_grouping_policy parameter, Configuration File Defaults, Multipaths Device Configuration Attributes, Configuration File Devices
- path_selector parameter, Configuration File Defaults, Multipaths Device Configuration Attributes, Configuration File Devices
- polling-interval parameter, Configuration File Defaults
- prio parameter, Configuration File Defaults, Configuration File Devices
- product parameter, Configuration File Devices
- product_blacklist parameter, Configuration File Devices
- queue_without_daemon parameter, Configuration File Defaults
- retain_attached_hw_handler parameter, Configuration File Defaults, Multipaths Device Configuration Attributes
- revision parameter, Configuration File Devices
- rr_min_io parameter, Configuration File Defaults, Multipaths Device Configuration Attributes
- rr_weight parameter, Configuration File Defaults, Multipaths Device Configuration Attributes, Configuration File Devices
- udev_dir parameter, Configuration File Defaults
- user_friendly_names parameter, Configuration File Defaults, Multipaths Device Configuration Attributes, Configuration File Devices
- vendor parameter, Configuration File Devices
- verbosity parameter, Configuration File Defaults
- wwid parameter, Multipaths Device Configuration Attributes

configuring
- DM-Multipath, Setting Up DM-Multipath

D

defaults section
- multipath.conf file, Configuration File Defaults

detect_prio parameter, Configuration File Defaults, Multipaths Device Configuration Attributes

dev/mapper directory, Multipath Device Identifiers

device name, Multipath Device Identifiers

device-mapper-multipath package, Setting Up DM-Multipath

devices
- adding, Configuring Storage Devices, Configuration File Devices

devices section
- multipath.conf file, Configuration File Devices

dev_loss_tmo parameter, Configuration File Defaults, Configuration File Devices

DM-Multipath
- and LVM, Multipath Devices in Logical Volumes
- components, DM-Multipath Components
- configuration file, The DM-Multipath Configuration File
- configuring, Setting Up DM-Multipath
- definition, Device Mapper Multipathing
- device name, Multipath Device Identifiers
- devices, Multipath Devices
- failover, Overview of DM-Multipath
- overview, Overview of DM-Multipath
- redundancy, Overview of DM-Multipath
- setup, Setting Up DM-Multipath
- setup, overview, DM-Multipath Setup Overview

dm-n devices, Multipath Device Identifiers

dmsetup command, determining device mapper entries, Determining Device Mapper Entries with the dmsetup Command

dm_multipath kernel module , DM-Multipath Components

F

failback parameter, Configuration File Defaults, Multipaths Device Configuration Attributes, Configuration File Devices

failover, Overview of DM-Multipath

fast_io_fail_tmo parameter, Configuration File Defaults, Configuration File Devices

features parameter, Configuration File Defaults, Configuration File Devices

features, new and changed, New and Changed Features

feedback
- contact information for this manual, We Need Feedback!

flush_on_last_del parameter, Configuration File Defaults, Multipaths Device Configuration Attributes, Configuration File Devices

force_sync parameter, Configuration File Defaults

G

getuid_callout parameter, Configuration File Defaults, Configuration File Devices

H

hardware_handler parameter, Configuration File Devices

hwtable_regex_match parameter, Configuration File Defaults

I

initramfs
- starting multipath, Setting Up Multipathing in the initramfs File System

K

kpartx command , DM-Multipath Components

L

local disks, ignoring, Ignoring Local Disks when Generating Multipath Devices

LVM physical volumes
- multipath devices, Multipath Devices in Logical Volumes

lvm.conf file , Multipath Devices in Logical Volumes

M

max_fds parameter, Configuration File Defaults

mpathconf command , DM-Multipath Components

multipath command , DM-Multipath Components
- options, Multipath Command Options
- output, Multipath Command Output
- queries, Multipath Queries with multipath Command

multipath daemon (multipathd), The Multipath Daemon

multipath devices, Multipath Devices
- logical volumes, Multipath Devices in Logical Volumes
- LVM physical volumes, Multipath Devices in Logical Volumes

multipath.conf file, Storage Array Support, The DM-Multipath Configuration File
- blacklist_exceptions section, Blacklist Exceptions
- defaults section, Configuration File Defaults
- devices section, Configuration File Devices
- multipaths section, Multipaths Device Configuration Attributes

multipath.conf.annotated file, The DM-Multipath Configuration File

multipath.conf.defaults file, Storage Array Support, The DM-Multipath Configuration File

multipathd
- command, Troubleshooting with the multipathd Interactive Console
- interactive console, Troubleshooting with the multipathd Interactive Console

multipathd daemon , DM-Multipath Components

multipathd start command, Setting Up DM-Multipath

multipathed root file system, Moving root File Systems from a Single Path Device to a Multipath Device

multipathed swap file system, Moving swap File Systems from a Single Path Device to a Multipath Device

multipaths section
- multipath.conf file, Multipaths Device Configuration Attributes

N

no_path_retry parameter, Configuration File Defaults, Multipaths Device Configuration Attributes, Configuration File Devices

O

overview
- features, new and changed, New and Changed Features

P

path_checker parameter, Configuration File Defaults, Configuration File Devices

path_grouping_policy parameter, Configuration File Defaults, Multipaths Device Configuration Attributes, Configuration File Devices

path_selector parameter, Configuration File Defaults, Multipaths Device Configuration Attributes, Configuration File Devices

polling_interval parameter, Configuration File Defaults

prio parameter, Configuration File Defaults, Configuration File Devices

product parameter, Configuration File Devices

product_blacklist parameter, Configuration File Devices

Q

queue_without_daemon parameter, Configuration File Defaults

R

resizing a multipath device, Resizing an Online Multipath Device

retain_attached_hw_handler parameter, Configuration File Defaults, Multipaths Device Configuration Attributes

revision parameter, Configuration File Devices

root file system, Moving root File Systems from a Single Path Device to a Multipath Device

rr_min_io parameter, Configuration File Defaults, Multipaths Device Configuration Attributes

rr_weight parameter, Configuration File Defaults, Multipaths Device Configuration Attributes, Configuration File Devices

S

setup
- DM-Multipath, Setting Up DM-Multipath

storage array support, Storage Array Support

storage arrays
- adding, Configuring Storage Devices, Configuration File Devices

swap file system, Moving swap File Systems from a Single Path Device to a Multipath Device

U

udev_dir parameter, Configuration File Defaults

user_friendly_names parameter , Multipath Device Identifiers, Configuration File Defaults, Multipaths Device Configuration Attributes, Configuration File Devices

V

vendor parameter, Configuration File Devices

verbosity parameter, Configuration File Defaults

W

World Wide Identifier (WWID), Multipath Device Identifiers
wwid parameter, Multipaths Device Configuration Attributes